PTO/SB/21 (01-08)
Approved for use through 01/31/2008. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Total Number of Pages in This Submission	

Application Number	09/911,663-Conf. #3836
Filing Date	July 24, 2001
First Named Inventor	John CIOLFI
Art Unit	2173
Examiner Name	N. Pillai
Attorney Docket Number	MWS-072

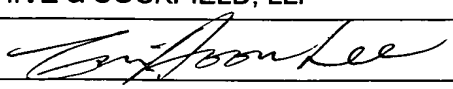
ENCLOSURES (*Check all that apply*)

- [ ] Fee Transmittal Form
  - [ ] Fee Attached
- [ ] Amendment/Reply
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [ ] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [ ] Certified Copy of Priority Document(s)
- [ ] Reply to Missing Parts/ Incomplete Application
  - [ ] Reply to Missing Parts under 37 CFR 1.52 or 1.53
- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition to Convert to a Provisional Application
- [ ] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, Number of CD(s) _____
  - [ ] Landscape Table on CD
- [ ] After Allowance Communication to TC
- [ ] Appeal Communication to Board of Appeals and Interferences
- [x] Appeal Communication to TC **(Appeal Notice, Brief, Reply Brief)**
- [ ] Proprietary Information
- [ ] Status Letter
- [x] Other Enclosure(s) (please Identify below):

Return Receipt Postcard

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	LAHIVE & COCKFIELD, LLP		
Signature			
Printed name	EuiHoon Lee		
Date	March 10, 2008	Reg. No.	L0248

Express Mail Label No. EM 194 129 602 US Dated: March 10, 2008

Express Mail Label No EM 194 129 602 US Dated: March 10, 2008

Docket No.: MWS-072
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
John Ciolfi

Application No.: 09/911,663 Confirmation No.: 3836

Filed: July 24, 2001 Art Unit: 2173

For: HANDLING PARAMETERS IN BLOCK DIAGRAM MODELING Examiner: N. Pillai

REPLY BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

As required under § 41.41(a), this brief is filed within two months of the Examiner's Answer mailed on January 9, 2008.

This brief contains items under the following headings:

I. Status of Claims

II. Grounds of Rejection to be Reviewed on Appeal

III. Argument

I. STATUS OF CLAIMS

Claims 33-44 and 46 are pending in the application. Claims 1-32 and 45 were canceled. Claims 33-44 and 46 are on appeal.

II. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 33-44 and 46 are rejected under U.S.C. §102(e) as being anticipated by United States Patent No. 6,937,257 B1 to Dunlavey ("Dunlavey").

III. ARGUMENT

A. Rejection of Canceled Claims under 35 U.S.C. §103(a)

In the Examiner's Answer of January 9, 2008, the Examiner rejected canceled claims 16-24, 26-32 and 45 under 35 U.S.C. §103(a) as being unpatentable over Dunlavey in view of United States Patent No. 5, 475, 851. (Examiner's Answer, pages 6-10).

Appellant specifically notes that an amendment canceling claims 16-24, 26-32 and 45 was filed with an Appeal Brief on June 8, 2007. Claims 33-44 and 46 are pending in the application and are on appeal. Appellant requests that the rejection of canceled claims 16-24, 26-32 and 45 under 35 U.S.C. §103(a) be withdrawn.

B. Rejection of Claims under 35 U.S.C. §102(e)

Further to the arguments presented in the Appeal Brief filed on June 8, 2007 and the Supplemental Appeal Brief filed on October 12, 2007, Appellant respectfully submits that Dunlavey fails to disclose ***pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters***, as recited in claims 33 and 46, for reasons set forth below.

In the Examiner's Answer, the Examiner refers to various portions of Dunlavey as disclosing the above feature of claims 33 and 46. Appellant submits that none of the portions of Dunlavey disclose the pooling feature recited in claims 33 and 46.

In the Examiner's Answer of January 9, 2008, the Examiner maintained her rejection of claims 33-44 and 46 under 35 U.S.C. §102(e) as being anticipated by Dunlavey. (Examiner's Answer, page 3).

Further to the arguments presented in the Appeal Brief filed on June 8, 2007 and the Supplemental Appeal Brief filed on October 12, 2007, Appellant respectfully submits that Dunlavey fails to disclose ***pooling together like non-interfaced run-time block parameters to reuse data for the like non-interfaced run-time block parameters***, as recited in claims 33 and 46, for reasons set forth below.

In the Examiner's Answer, the Examiner refers to various portions of Dunlavey as disclosing the above feature of claims 33 and 46. Appellant submits that none of the portions of Dunlavey disclose the pooling feature recited in claims 33 and 46.

1. Multidimensional Data Type

In the Examiner's Answer, the Examiner alleges that "[t]he multidimensional data type is one example of a variable that is a combination of pooled together like run time block parameters." (Examiner's Answer, page 11). Appellant respectfully disagrees.

Appellant respectfully submits that the multidimensional data type in Dunlavey does not anticipate the feature of ***pooling together like non-interfaced run-time block parameters***, as recited in claims 33 and 46, because the multidimensional data type does not pool like run-time parameters in Dunlavey.

1.1. Multidimensional Data Type Does Not Pool Like Parameters

In claims 33 and 46, a plurality of user-defined block parameters are processed to produce a plurality of run-time block parameters. *Like* non-interfaced

run-time block parameters are *pooled* together. As discussed in the Appeal Brief filed on June 8, 2007 and the Supplemental Appeal Brief filed on October 12, 2007, a single common representation may be used for the pooled like non-interfaced run-time block parameters. For example, a single entry may be maintained in a run-time parameter table and used for the pooled like non-interfaced run-time block parameters. (Appeal Brief, page 5 and Supplemental Appeal Brief, page 5).

In comparison, the multidimensional data type in Dunlavey is a data type that represents variables in the internal format of a model. In Dunlavey, instead all variables are represented using the multidimensional data type in the internal format. Dunlavey does not use the multidimensional data type to pool only like variables. In Dunlavey, any variables in a model are represented using the multidimensional data type in the internal format of the model.

1.2. Multidimensional Data Type Is Not Used To Pool Run-time Parameters

Furthermore, the multidimensional data type of Dunlavey is not used to represent run-time block parameters. In Dunlavey, the multidimensional data type is used in the internal format of a model to identify inconsistency in the units of variables in the model before the model is executed. Dunlavey uses the multidimensional data type in the internal format of a model, but not in the model execution time. There is no discussion in Dunlavey of using the multidimensional data type in the model execution time. There is no discussion in Dunlavey that the multidimensional data type pools together *run-time* parameters that are used in the model execution time.

The Applicants find it hard to understand the Examiner's argument asserting that the multidimensional data type of Dunlavey corresponds to run-time

parameters. The Examiner states that: "This testing of the model using real-time emulation discloses that the parameters are being executed as the parameters are defined by the user [sic]. The parameters take on an internal format and become run-time parameters so that the model generated can be tested. This suggests that execution is occurring as the user is defining the parameters." (Examiner's Answer, page 11). However, nothing in Dunlavey implies that the model execution is happening as the user is defining the parameters, and even if it were to be so, the Applicant is perplexed as to how it would have a bearing on whether the multidimensional data type is a run-time parameter.

To the best of the Applicant's understanding, the Examiner is arguing that, since the model of Dunlavey is executed, the parameters in it are necessarily run-time parameters from the time they are defined. However, Dunlavey itself teaches away from that and describes how to convert multi-dimensional data types into high-level language source code that may be used for execution. (Dunlavey, column 19, lines 45-47). Therefore, nowhere does Dunlavey teach or suggest that the multidimensional data types are run-time parameters that may be pooled together.

2. SetDiscrete

The Examiner also alleges that "[t]he 'SetDiscrete' variable includes multiple variables that are pooled together and are jointly distributed where multiple variables are represented in the one variable 'SetDiscrete.'" (Examiner's Answer, page 11). Appellant respectfully disagrees.

Appellant respectfully submits that the SetDiscrete primitive in Dunlavey does not anticipate the feature of ***pooling together like non-interfaced run-time block parameters***, as recited in claims 33 and 46, because the SetDiscrete primitive does not pool like run-time parameters in Dunlavey.

2.1. SetDiscrete Does Not Pool Like Parameters

In Figure 4, Dunlavey describes the SetDiscrete as one of internal data structures. In Dunlavey, the SetDiscrete primitive is used to set a group of categorical variables that are jointly distributed. In Dunlavey, the term "distributed" is used as a mathematical term describing a probability function associated with values that a variable may take on. For example, the values may be distributed according different probability distribution functions, such as normal distribution, poisson distribution, etc. The term "distributed" as used by Dunlavey has nothing to do with how or where the parameters are stored or used within the system. The Applicants have raised this argument before, and the Examiner has not supplied any evidence or citation to Dunlavey that would point to a contrary interpretation of what is represented by the SetDiscrete primitive in Dunlavey.

The Examiner glosses over the term "*like* ...parameters" used in claims 33 and 46. In attempting to find a passage in Dunlavey disclosing "like parameters," the Examiner states that: Dunlavey teaches "a group of run-time block parameters that are grouped and share a commonality of belonging to this group." (Examiner's Answer, page 4). However, "belonging to the same group" does not imply that the grouped parameters are alike. Any number of dissimilar parameters may be grouped together. Indeed, if the term "like parameters" were to be interpreted as "parameters belonging to the same group," it would render meaningless the additional limitation of *"grouping* like... parameters." Therefore, the Applicants contend that the term "like... parameters" should not be interpreted as "parameters sharing a commonality of belonging to the same group," and that Dunlavey does not teach or suggest pooling together like non-interface run-time block parameters.

Dunlavey clearly discloses that "any set of variables can be joined into a joint multivariate distribution (emphasis added)." (Dunlavey, column 8, lines 50-51). That is, the variables in the group set by the SetDiscrete primitive in Dunlavey can be any set of variables. The SetDiscrete primitive in Dunlavey does not *pool* together *like* non-interfaced run-time block parameters.

2.2. SetDiscrete Does Not Pool Run-time Parameters

Furthermore, the variables that are grouped using the SetDiscrete primitive in Dunlavey are not run-time block parameters. The SetDiscrete primitive is a data structure used in the internal representation of a model in Dunlavey. This internal representation is later converted to high-level language source code, which is subsequently compiled and linked to execute. (Dunlavey, column 19, lines 45-47). In high-level language source code, all variables are defined as global variables and assigned names that do not conflict with any previously assigned variable names. (Dunlavey, column 19, lines 56-65). The variables in the group set using the SetDiscrete primitive do not appear in the high-level language source code and cannot be construed as *run-time* block parameters.

3. Categorical Distribution Block and Multivariate Distribution Block

The Examiner further alleges that "categorical distribution block and multivariate distribution block both represent a single representation that is used to pool together multiple variables across time." (Office Action, page 12). Appellant respectfully disagrees.

Appellant respectfully submits that the categorical distribution block and the multivariate distribution block in Dunlavey do not anticipate the feature of ***pooling together like non-interfaced run-time block parameters***, as recited in claims 33

and 46, because the categorical distribution block and the multivariate distribution block do not pool non-interfaced parameters.

3.1. Categorical Distribution Block and Multivariate Distribution Block Do Not Pool Non-interfaced Parameters

The categorical distribution block in Dunlavey "represents a quantity that is known to be variable over time, and for which the number of possible different values is finite and small (emphasis added)." (Dunlavey, column 8, lines 5-7). The multivariate distribution block in Dunlavey "represents multiple quantities that are known to be variable over time." (Dunlavey, column 8, lines 10-11). The categorical distribution block and the multivariate distribution block do not pool "non-interfaced" run-time block parameters – that is, parameters that are not variable during the execution. Dunlavey clearly describes that the categorical distribution block and the multivariate distribution block represent a variable quantity or quantities. Dunlavey does not disclose "pooling together like *non-interfaced* run-time block parameters (emphasis added)," as recited in claims 33 and 46.

The Examiner alleges that "[i]n this example, Dunlavey has clearly disclosed that the variables cannot be modified by the user (column 10, lines 5-15)." (Examiner's Answer, page 12). The Examiner's interpretation of the "non-interfaced" parameters as variables that are not modified by the user is not consistent with the specification of the present application. The present application describes non-interfaced parameters as the parameters that "remain constant during model execution or do not change in the generated code." (Specification, page 11, lines 14-16). Dunlavey does not disclose "pooling together like *non-interfaced* run-time block parameters" that "remain constant during model execution."

4. Adjustment to Variables

The Examiner refers to Dunlavey, column 25, lines 40-51 and alleges that "Dunlavey does disclose providing means for the user to change values associated with variables where this involves reuse of variables that are in the graphical model including pooled block variables which are representations of multiple variables," and "[b]ased on the user adjustment to variables the same parameters can be used to calculate new results." (Examiner's Answer, pages 12-13). Appellant respectfully disagrees.

Appellant respectfully submits that the adjustment of variables in Dunlavey doe not anticipate ***pooling ... to reuse data for the like non-interfaced run-time block parameters***, as recited in claims 33 and 46, because the adjustment of variables does not involve reuse of data for the pooled parameters in Dunlavey.

4.1. Adjustment to Variables Does Not Involve Reuse of Data for Pooled Parameters

Dunlavey, at column 25, lines 40-51, recites:

> The values for the various parameters can be changed by entering new values in the value field of the parameters portion 750 , or by clicking the arrow icons or clicking and dragging the wheel adjuster in the value adjustment portion 754 of the debug window 740 . Whenever an adjustment is made to one of the parameters, the graph in the graphing portion 742 of the debug window 740 is updated substantially immediately thereafter. In one embodiment, this graph update is accomplished automatically because the graph is continuously being updated by active interpretation of the internal format statements by the simulation interpreter 144 , even when the parameters remain unchanged.

In the above quoted portion, Dunlavey describes that a user may change the values of various parameters to debug a model. Dunlavey also discusses updating a graph automatically with or without changing the values of parameters. Dunlavey, however, does not disclose that changing the values of parameters invloves reusing data for the pooled parameters.

Appellant respectfully submits that the Examiner's allegation is based on misinterpretation of ***pooling ... to reuse data for the like non-interfaced run-time block parameters***, which is recited in claims 33 and 46. The Examiner alleges that the same parameters are used to calculate new results. The Examiner appears to allege that the same parameters are used in the iterations of execution of a model to calculate new results. Reuse of the same parameters in the iterations of execution of a model is not reuse of data for the pooled parameters. In the reuse of data for the pooled parameters, a single representation may be reused for the pooled like parameters during execution of a model. The reuse of the same parameters to calculate new results is not the reuse of data for the pooled parameters. There is no reuse of data for the pooled parameters in Dunlavey.

5. Global Variable

The Examiner also refers to Dunlavey, column 10, lines 5-15 and alleges that "these include global variables which have one instance but can be retrieved and reused multiple times within a procedure," and "[t]he purpose of global variable is to provide one definition, with the variable being accessible to multiple procedures and therefore reused multiple times." (Examiner's Answer, page 13). Appellant respectfully disagrees.

Appellant respectfully submits that a global variable in Dunlavey does not anticipate ***pooling ... to reuse data for the like non-interfaced run-time block***

parameters, as recited in claims 33 and 46, because the global variable does not reuse data for the pooled parameters.

5.1. A Global Variable Does Not Reuse Data for Pooled Parameters

Dunlavey, at column 10, lines 5-15, recites:

> As each variable is added to a procedure block, the variable becomes an output of the block. The only variables that may be set in a procedure block are the local procedure and integrator variables, although global variables can be referenced, and the local procedure variables are all non-static. These limitations are placed on the procedure block to avoid the problems-that could otherwise be created by the user. For example, if a user were to change a global variable inside a procedure block, there may be too much opportunity for human error to introduce bugs into the trial simulation, possibly reducing the benefits of the invention.

In this portion, Dunlavey discusses a procedure block in which local procedure variables are set and global variables are referenced. Dunlavey, however, does not disclose reusing data for the like non-interfaced run-time block parameters. Although Dunlavey describes that a global variable can be referenced in a procedure block, the reference to the global variable by the procedure block is not reuse of data for pooled parameters.

The Examiner alleges that the global variable is accessible to multiple procedures and reused multiple times. In the Examiner's allegation, a single global variable is used multiple times. The Applicant is unable to ascertain the relevance of that assertion. Even if the purpose of the global variable were to be similar to the purpose behind pooling like non-interfaced run-time parameters, *arguendo*, it would in no way suggest that the former anticipates the later. In

Dunlavey, the same global variable is used or referenced in multiple procedures. The global variable is not reused for the pooled multiple parameters.

For at least the reasons set forth above, Appellant respectfully submits that Dunlavey fails to disclose each and every element of claims 33 and 46. Claims 34-44 depend on claim 33 and, as such incorporate the subject matter of claim 33. Therefore, Appellant respectfully requests that the 35 U.S.C. §102(e) rejection of claims 33-44 and 46 be reversed and claims 33-44 and 46 be allowed.

Dated: March 10, 2007

Respectfully submitted,

By 

EuiHoon Lee
Registration No.: L0248
LAHIVE & COCKFIELD, LLP
One Post Office Square
Boston, Massachusetts 02109-2127
(617) 227-7400
(617) 742-4214 (Fax)
Attorney/Agent For Appellant